Object-Oriented Design

Object-Oriented Concepts
'Class'

1

Basic Knowledge

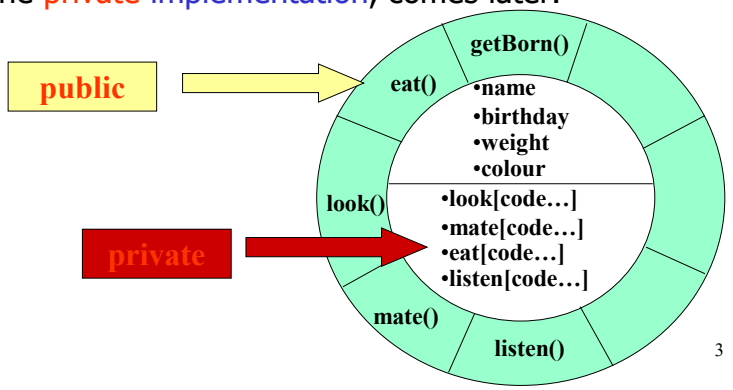
Object-oriented concepts

- ✓objects: state, behaviour, identity
- ✓relationships between objects
- ✓classes: inside, outside (public interface)
- ✓class relationships: inheritance, using, instantiation
- ✓(class) attributes, (class) methods – operations

2

Object-oriented concepts : classes: inside, outside (public interface)

- the **outside** is the interface with external world = the public behaviour, is modelled first
- the **inside** hides the implementation: properties (attributes) and behaviour (coded in the methods), the private implementation, comes later.



3

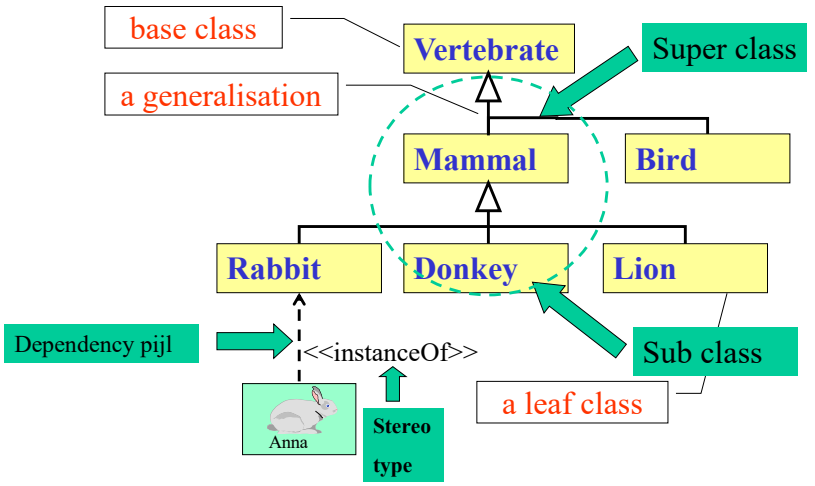
Object-oriented concepts : class relationships: inheritance, using

Different types of **relationships between classes**:

- **Inheritance** relationships: a kind-of relationship (**is** an ….)
- **Associations**: a using relationship (**has** an ….)
  - A special kind of an association: an **aggregation** or a part-of relationship
  - A special kind of an aggregation: a **composition**

4

Object-oriented concepts : class relationships: inheritance



5

Object-oriented concepts : class relationships: inheritance

- a donkey is a kind of mammal
- a rabbit is also a kind of mammal
- a bird is not a mammal but it has common characteristics and common behaviour with mammals, they are both vertebrates.

...

6

Object-oriented concepts : class relationships: inheritance

Specialisation and **generalisation** are two ways to build an inheritance relationship.

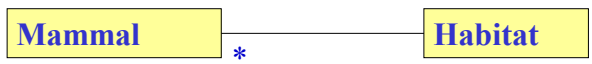
1.Generalisation is the process in which we start with different classes that have similar behaviour and similar characteristics and decide to merge the common behaviour and the common characteristics in a super class

2.Specialisation is the process in which we start with one class and decide to split up specific behaviour and specific characteristics in different subclasses

7

Object-oriented concepts : class relationships: using -> association

- A mammal shares its habitat with other mammals.
  = a using relationship



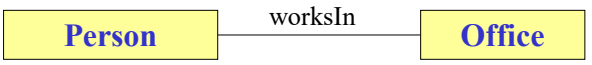
This diagram says that <u>many</u> (= 0 or more) mammals are connected to (share) a habitat.

8

Object-oriented concepts : class relationships

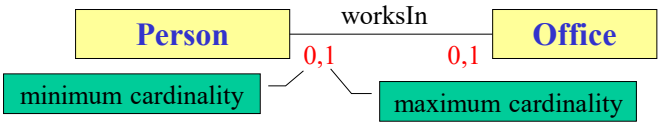
implementation of an association with 1-1 cardinality

- A association without further indication means that the maximum cardinality is one and the minimum cardinality is zero.



This diagram says that a person works in only one office and that in the office there is only one person. Possibly a person may not have an office or office may be empty (have no person that is connected with the office).

We can also explicitly show the cardinality.

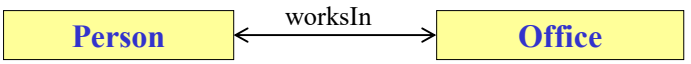


9

Object-oriented concepts : class relationships

implementation of an association with 1-1 cardinality

- The association is implemented through an attribute.



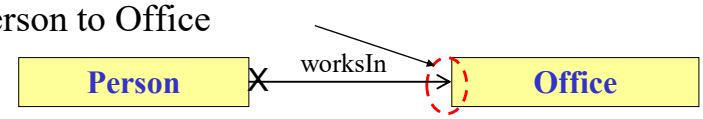
1. "office" is an attribute of type Office. It realises the association between Person and Office in the direction of the class Office.
2. The attribute "person" in class Office realises the association in the reverse direction.

10

Object-oriented concepts : class relationships

implementation of an *directed* association with 1-1 cardinality

- When the association is **directed** (restricted) from Person to Office



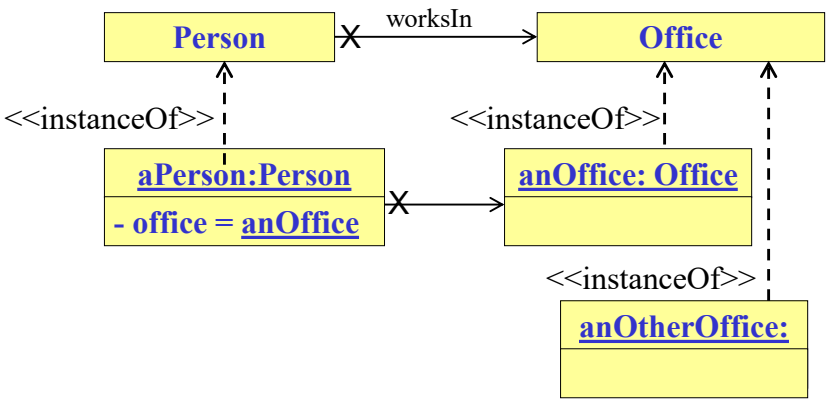
Only the attribute "office" remains.

11

Object-oriented concepts : class relationships

implementation of a *directed* association with 1-1 cardinality in the objects

- The office attribute of the object aPerson holds the value anOffice.

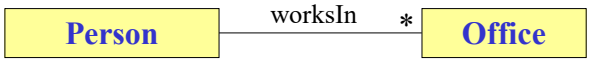


12

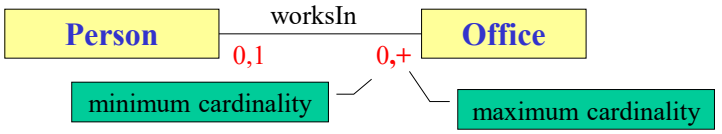
Object-oriented concepts : class relationships

implementation of an association with 1-many cardinality

- A association with a many (*) cardinality indicates that different instances of the class at the many side can be associated with a single instance from the class at the other side of the association. The minimum cardinality remains zero.



This diagram says that a person can work in different offices but that in an office there is only one person. Possibly a person may not have an office or office may be empty (have no person that is connected with the office). We can also explicitly show the cardinality.

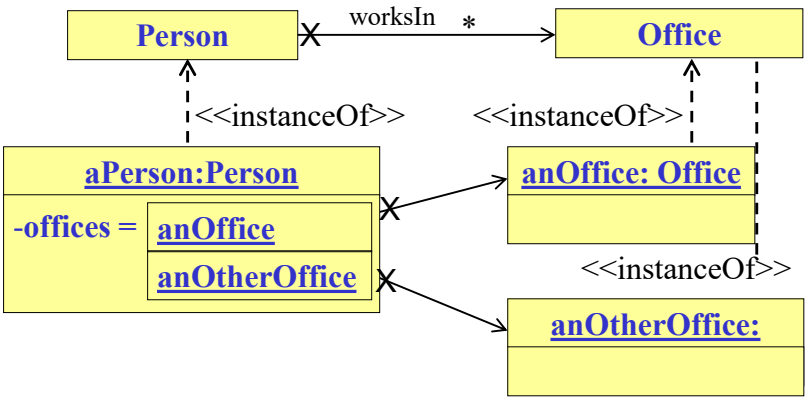


13

Object-oriented concepts : class relationships

implementation of a *directed* association with 1-many cardinality in the objects

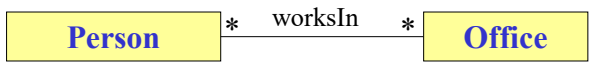
- The office attribute of the object aPerson holds the values anOffice and anOtherOffice.



Object-oriented concepts : class relationships

implementation of an association with many-many cardinality

- A association with a many (*) cardinality indicates that different instances of the class at the many side can be associated with different instances from the class at the other side of the association.



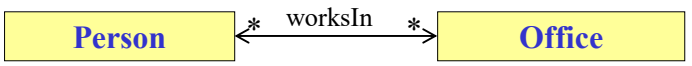
This diagram says that a person can work in different offices and that an office can be occupied by many persons. Possibly a person may not have an office or office may be empty (have no person that is connected with the office).

15

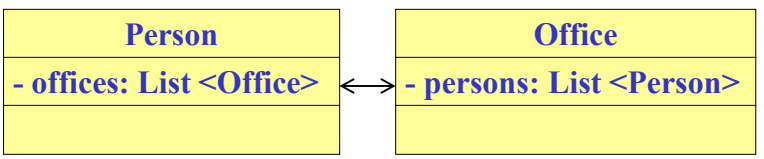
Object-oriented concepts : class relationships

implementation of an association with many-many cardinality

The association is implemented through "list" attributes.



"offices" is an attribute in the class Person that holds a list of offices. The attribute "persons" in class Office is a list of persons working in that office.

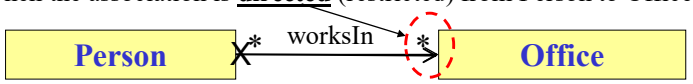


16

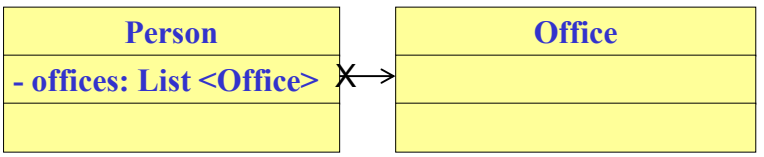
Object-oriented concepts : class relationships

implementation of an *directed* association with *-* cardinality

- When the association is **directed** (restricted) from Person to Office



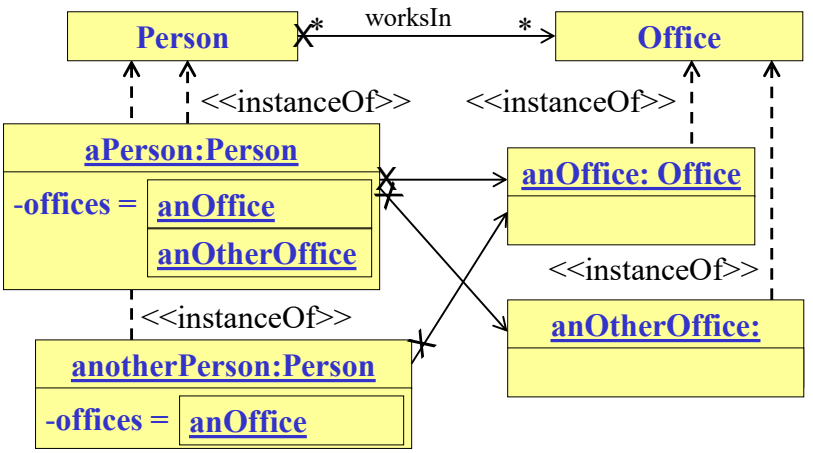
Only the attribute "offices" remains.



17

Object-oriented concepts : class relationships

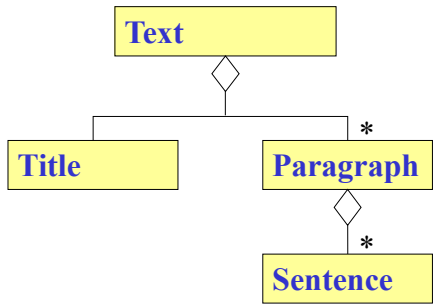
implementation of an *directed* association with many-many cardinality in the objects



18

Object-oriented concepts : class relationships
aggregation

Part-of relationships show a connection between a whole and its parts. Aggregations are a special form of **associations** , whereby a part does not depend on the whole to exist.

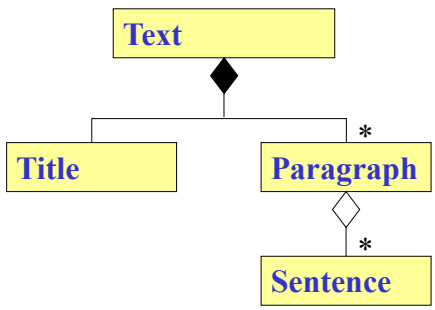


19

Object-oriented concepts : class relationships
composition

Part-of relationships show a connection between a whole and its parts. Compositions are a special form of aggregations, whereby the parts depend on the whole to exist.

Notice that the same problem domain can be either seen as an aggregation or a composition!



20

Object-oriented concepts : class relationships
Inheritance and instantiation

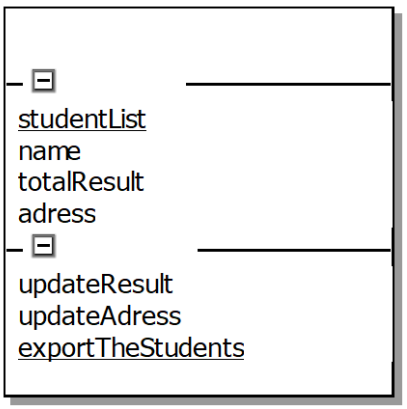
- Classes have two types of **descendants**:
  - **instances**
  - **sub-classes**
- An **abstract class** is a class that cannot (is not allowed to) have instances
- A **concrete class** can have instances

An abstract Class, therefore, will always have subclasses (concrete or abstract). Only a concrete class can act as a leaf in the class hierarchy..

Whether a class is modelled as concrete or abstract depends on the viewpoint.

21

Object-oriented concepts : (class)attributes, (class)methods-operations



22